Applying Model-based SE Techniques for Dependable Land Systems

Payne, R, Fitzgerald, J, Bryans, J & Winthorpe, E

Original citation & hyperlink:
Payne, R, Fitzgerald, J, Bryans, J & Winthorpe, E 2016, 'Applying Model-based SE Techniques for Dependable Land Systems' pp. 1783–1798. DOI: 10.1002/j.2334-5837.2016.00261.x
https://dx.doi.org/10.1002/j.2334-5837.2016.00261.x

DOI 10.1002/j.2334-5837.2016.00261.x

Publisher: Wiley

26th Annual INCOSE International Symposium (IS 2016)
Edinburgh, Scotland, UK, July 18-21, 2016

Applying Model-based SE Techniques for Dependable Land Systems

Richard Payne, John Fitzgerald
Newcastle University, UK
richard.payne@ncl.ac.uk
john.fitzgerald@ncl.ac.uk

Jeremy Bryans
Coventry University, UK
jeremy.bryans@coventry.ac.uk

Elspeth Winthorpe
MOD, UK
elspeth.winthorpe100@mod.uk

Abstract. A System of Systems (SoS) is a group of independent constituent systems that by their interactions together deliver an emerging capability on which reliance is placed. In the UK Ministry of Defence (MOD) Land domain, the appetite for risk caused by uncertainty of SoS dependencies is particularly low. Dependability and assurance are therefore vital, both in the integration and interoperability of individual constituent systems, and at the level of the SoS as a whole. This paper presents the findings from a study, asking whether model-based engineering technologies could potentially deliver a pragmatic method for Land Open Systems Arrchitecture (LOSA) SoS verification and assurance. We conclude that existing model-based SoS and Cyber-Physical Systems engineering techniques could potentially deliver a basis for LOSA SoS requirement engineering, design, verification and assurance. However, the techniques studied are at varying levels of maturity and therefore we identify areas of future work that could inform the ongoing research and experimentation programmes for LOSA and others across defence.

1. Introduction

A wide range of capabilities in both civil and defence domains are delivered by the collective operation of systems that are independent of one another. For example, successful completion of a military operation may involve multiple units from different services all with their own power and data networks, logistical systems, vehicles and units that may have evolved separately and may not have been conceived with collective operations in mind at all. As reliance is placed on the performance of these groups of systems, they merit consideration as systems in their own right (Maier 1998). Such systems are termed Systems of Systems (SoSs), and their successful creation, integration and maintenance presents considerable challenges (Dahmann 2014).

Dependability is of vital importance both at the SoS level and in the integration and interoperability of the individual Constituent Systems (CSs) making up the SoS. It is therefore vital that stakeholders involved in system requirements, design, verification and assurance have evidence that increases their understanding of the interdependencies in an SoS, and thus confidence in trade-off decisions. In Land Open System Architecture (LOSA) – an approach to develop, ensure and assure coherence across the Land Domain – this requirement is particularly acute in the development of safety and reliability cases where the appetite for risk caused by uncertainty of SoS dependencies is particularly low.

Challenges in SoS engineering (SoSE) arise due to the limited information available about CSs, the need for clear descriptions of interactions between them, and the need for rigorous analysis

of the SoS behavior that emerges from these interactions. Model-based techniques are increasingly seen as a way of mitigating these complexities by encouraging the precise description of SoS architectures and CS interfaces, as well as providing means for systematic and possibly automated analysis of dependability properties.

The contribution of this paper is to establish the feasibility of an approach proposed by MOD stakeholders that uses model-based SoS engineering (MBSoSE) as a means of providing stakeholders with the evidence needed to increase confidence in the LOSA programme. We establish this feasibility by assessing a set of design-time model-based techniques using an illustrative LOSA example with dependability requirements.

In the remainder of the paper, we provide an overview of related work, outline and illustrate the collection of technologies used and we conclude by considering their applicability and maturity for LOSA.

2. Related Work

We build on related work in three areas: model-based SoS engineering, dependability in SoSs and open systems in the defence domain.

The systems engineer working in an SoS context faces challenges stemming from the lack of centralised authority over constituents, the complexity of predicting emergent behaviours, and the diversity of stakeholders (INCOSE, 2014). In common with other areas of Systems Engineering, there has been considerable interest in the use of model-based techniques as a way of addressing these challenges (Cantot and Luzeaux, 2011), (DoD, 2008). Recent EU projects have developed model-based technologies aiming to address SoSE. Both the COMPASS[1] and DANSE[2] projects piloted guidelines, patterns and tools for SoS modelling and verification (Andrews et al. 2014, Bryans et al. 2014, Ingram et al. 2015). Recent projects have also addressed multi-criticality (AMADEOS [3]), local/global optimisation (LOCAL4GLOBAL[4]), and dynamic management of physically coupled SoSs (DYMASOS[5]). Roadmaps for research and innovation in Europe (Road2SOS[6] and CPSoS[7]) have emphasised the need for these developments. One of the most comprehensive reviews of the field (Nielsen et al., 2015) argues for modelling languages and methods able to describe the interoperability of networked, distributed constituent systems (one of the key motivations for open architectural approaches such as LOSA), and support simulation, with well-founded ("formal") semantics to enable generation and maintenance of tests.

The field of systems dependability has been a subject of substantial study (Avizienis et al. 2004) but its extension to systems of systems brings new challenges. Chief among these is the necessity to define and validate dependability in the face of the emergent behaviour that characterizes systems of systems (Maier 1996) (Kopetz et al. 2015). Many approaches to SoS dependability have built on approaches to system dependability, and contractual styles similar to the one we propose here are popular – for example (Bryans et al. 2013), (Benveniste et al.,

[1] http://www.compass-research.eu/
[2] www.danse-ip.eu
[3] www.amadeos-project.eu
[4] http://local4global-fp7.eu/
[5] www.dymasos.eu
[6] www.road2sos-project.eu
[7] www.cpsos.eu

2012), (Damm et al. 2011). These trace their origins to the Design by Contract approach introduced by Meyer (Meyer 1998).

Within MOD, open systems are characteristically composed of modular cooperative components from several sources, the system is also not dependent upon any proprietary elements (Defence Committee UK. 2013). In recent years there has been a push within the MOD to achieve greater interaction between different systems in order to improve operational effectiveness across all Defence (MOD 2013b). Interoperability is a constant issue within defence, open systems enable the use of equipment which was procured at different times and contractors to be able to have "*plug and play*" functionality, allowing more capacity during upgrading and enables easing of interoperability with new capabilities (MOD 2012b).

3. Approach to Model-based SoS Engineering

This paper examines a sequence of design-time model-based techniques that may be used to implement a sequence of activities proposed by MOD stakeholders, form the basis of a pragmatic method to enable the assessment of dependability in the context of LOSA. Few if any single existing methods are able to deliver the design needs of a LOSA SoS while satisfying the requirements of stakeholder groups. We therefore consider several techniques, using notations and methods from the SoSE and CPS engineering (CPSE) disciplines.

To undertake SoSE, we used SysML (OMG 2012, Holt et al. 2014), a semi-formal modelling language supported by industrial strength tools, and the experimental formal language CML (Woodcock 2012), designed during the recent SoS project COMPASS and supported by the prototype Symphony tool platform (Coleman et al. 2014). To carry out CPSE, we used the Crescendo tool (Fitzgerald et al.2014), which uses the languages VDM (Fitzgerald et al. 2009) and 20-sim[8]. Given a sequence of design activities defined by MOD, we use these techniques as detailed below, and summarized in Table 1.

1. **Clarify and formally define the relationships between architectures, standards and implementations**. We use a model-based ontology (Holt et al. 2014) to identify concepts in LOSA; including standards, implementation details and modelling terms.
2. **Understanding stakeholder roles to allow definition of what LOSA compliance means to stakeholders [...]**. The study demonstrates a requirements engineering technique developed for SoS (Holt et al. 2012) (applied in (Ingram et al. 2015) and (Andrews et al. 2014)) to identify stakeholders and place LOSA SoS requirements in context. This technique may be augmented with traceability links throughout an architectural model.
3. **Define the architectural framework representations required in support of stakeholder roles [...] including formal clarification of system boundaries within LOSA**. The use of architectural modelling techniques, and in particular a pattern for defining Interface Contracts (ICs) (Bryans et al. 2014), that allows an SoS to be defined in terms of its CSs. Defining the interfaces between the CSs enables the explicit description of CS interactions. Defining these interfaces together with the SoS enables the clarification of the SoS boundary.
4. **[...] enable assessment and measurement of the dependencies between system boundaries to deliver evidence and confidence of the understanding of SoS behaviors**

[8] http://www.20sim.com/

and limitations to inform safety and reliability cases. The model-based techniques employed (SysML and CML) have associated analysis technologies which produce results we can use. The analysis technologies we used in this paper (formal model simulation (Coleman et al. 2014), fault modelling (Andrews et al. 2014) and co-simulation (Fitzgerald et al. 2014)) vary in their ease of use, and offer varying degrees of confidence to the user.

5. **Develop generic reference models for platform interoperability and platform integration**. A contractual style of architectural modelling, advocated in the IC pattern, is a candidate for defining reference models, useful in integration and reasoning about interoperability. The pattern encourages the definition of an SoS model in terms of the assumptions and guarantees which are placed on the CS, enabling the analysis of integrating new CSs into the SoS. By defining the SoS contractually, the CSs which form the SoS must conform to those ICs – and may do this in which ever way they choose.
6. **Use models to deliver evidence and confidence for use in safety and reliability case developments as required throughout the lifecycle**. The models and analysis results may be used to provide evidence and confidence to an SoS engineer. Using these as part of safety and reliability cases requires further investigation.

Table 1: Techniques applied to sequence of MOD LOSA challenges

Challenge	Applicable Technique	Technology Used
1	Model-based Ontology	Ontology Definition Viewpoint
2	Context-based Requirements Engineering	SoS-ACRE
3	Architectural Modelling	Interface Contract Pattern
4	Model Analysis (formal model simulation, fault modelling and co-simulation)	CML model simulation, Fault Modelling Architectural Framework and Crescendo co-simulation
5	Architectural Modelling	Interface Contract Pattern
6	Models and analysis results	

4. Illustrative Example

To demonstrate feasibility, we develop an illustrative example architecture, and a subset of the dependability concerns typical in a LOSA SoS. The purpose of the illustrative example is for us to define the scope of interest in this paper, and therefore we abstract from concrete implementation details. For the purposes of this study, we make some simplifications, as shown in the SysML Block Definition Diagram (BDD) in Figure 1. The example SoS comprises a *Mounted Soldier*, a *Dismounted Soldier*[9], a *Vehicle* and a *Patrol Officer*. For the purpose of this example, we treat these four elements as individual CSs.

We employ an established definition of dependability (Avizienis et al. 2004) that divides the topic into: the attributes which comprise dependability; the threats to dependability; and the means that may be employed to ensure a system's dependability. In this study, rather than contrive an example to cover the full range of dependability properties, we have concentrated on a selected subset, in particular *availability*, *reliability* and *safety*.

[9] A soldier is classed as *mounted* if they are a member of the vehicle's crew, otherwise they are *dismounted*.

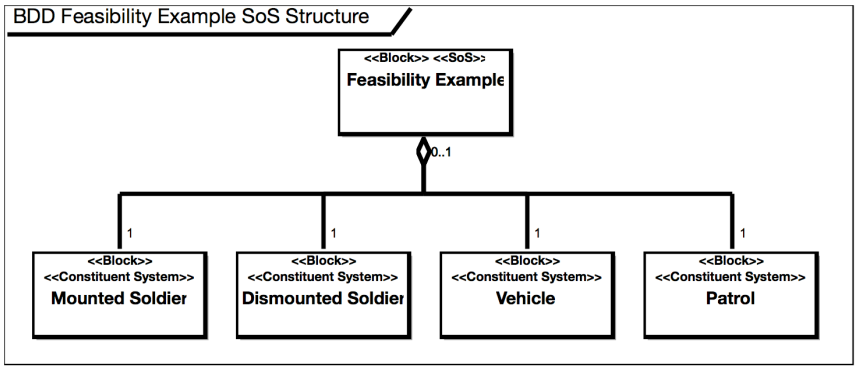

Figure 1: Block Definition Diagram showing composition of the example SoS

In a LOSA SoS, we have CSs with both digital and physical phenomena, and so we propose dependability properties related to computation/networks and physical issues. Beginning with discrete conditions and transitions of the feasibility example, we identify a set of three representative dependability properties:

- Availability: What should we expect if the high bandwidth interface is not available?
- Reliability: What if the low bandwidth channel loses a message? Are the individual logs recorded by the Soldiers and the Vehicle consistent?
- Safety: Can we distinguish between safe and unsafe states?

The next set of properties are in terms of a continuous variable, namely electrical power. This allows us to explore the techniques available to answer questions related to continuous time. We consider only availability and reliability in the context of continuous time properties:

- Availability: What is the system behaviour if power goes below 9v?
- Reliability: If power goes below 9v, will mission logs remain equal?

In the remainder of this section, we present models developed, and analysis performed using those technologies defined in Table 1, using the illustrative example where relevant. Several technologies have been realized in previous work, and therefore for brevity, we summarize those efforts here, providing references to those works for interested readers.

4.1 LOSA Ontology

We begin by explicitly defining the relationships between concepts LOSA, SoSs and the IC Pattern in an *ontology* presented as an Ontology Definition View (ODV) in Figure 2. The ODV is a view from the COMPASS AF Framework (Holt et al. 2015), a framework for the definition of architectural frameworks.

The ODV includes three main categories: Defence Standards (DefStans) (blue); Land domain concepts (green); and SoSs and ICs (brown). Relationships between the concepts are defined, for example; a *Vehicle Platform* conforms to the *Generic Vehicle Architecture* standard. Of importance in this work is the correspondence between MOD terms and those used in the SoSE domain. In the ODV, we state that a *Land Deployment* corresponds to a *System of Systems* and the different platforms correspond to *Constituent Systems*, which expose *COIL Interface*s. The ODV states that a *CS* may conform to several *ICs* each of which also expose *COIL Interfaces*. The composition of a collection of *ICs* is defined as an *IC-defined SoS*. The *SoS* conforms to an *IC-defined SoS*. Finally, an *SoS-level Contract* governs the *IC-defined SoS*.

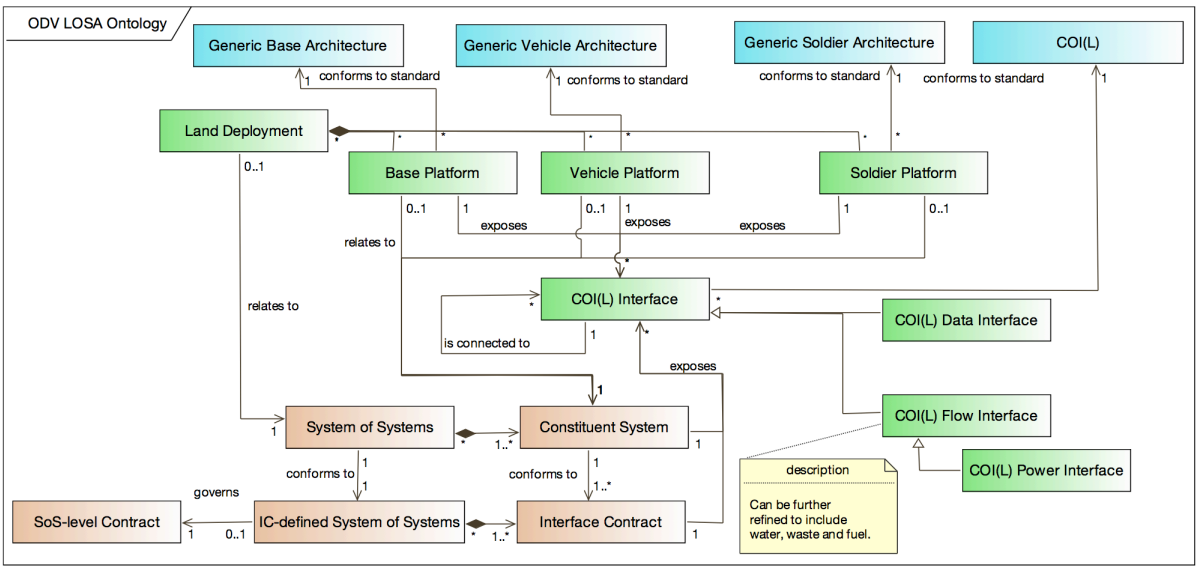

Figure 2: Ontology Definition View for LOSA concepts

4.2 Interface Contracts

We propose the use of modelling patterns and frameworks when defining LOSA architectural models. The first – the Interface Contract pattern – promotes a contractual approach to defining the CSs in terms of the obligations and reliances they may place on the other members of the SoS. The IC pattern is intended to allow the SoS engineer to specify limited internal behaviours to which CSs are required to conform, in order for them to a part of the SoS. CSs may conceivably conform to multiple ICs (and not necessarily ICs of the SoS of interest), and may conform to the IC in any way they wish. Defining the SoS using this pattern allows the engineer to analyse, and maintain, global emergent properties under SoS evolution and integration.

The IC pattern, however, is focused on the data-based phenomena of SoSs. It enables the definition of data, operations to change data and the means to describe communications. Therefore, we extend the IC pattern to consider how flow-based properties may be modelled and used to enrich the contractually defined CSs. In LOSA SoSs the interfaces include: power, fuel, water and waste. The interfaces required differ between the different CS types.

In this project we consider the feasibility of modelling such flows along with the contractual description for structure and behavior of ICs and extend the pattern to model:

- Power-related values and inputs/outputs of CSs (extends the IC Identification View);
- Flow of power between CSs (extends the IC Connections View (ICCV));
- Constraints based on physical properties (extends the IC Protocol Definition View);
- Relationships between power-related values (incorporates use of Parametric Diagrams).

In Figure 3 we show the modified ICCV, which now includes connectors with item flows, and depicts the flow of power from the *Vehicle IC* to the *Mounted-Soldier Soldier IC*. In addition, we demonstrate the use of constraints on external power corresponding to requirements in the Generic Soldier Architecture standard (MOD 13a). Whilst this may be better suited as a uniquely defined requirement, we feel that representing this directly on the ICCV is also useful.

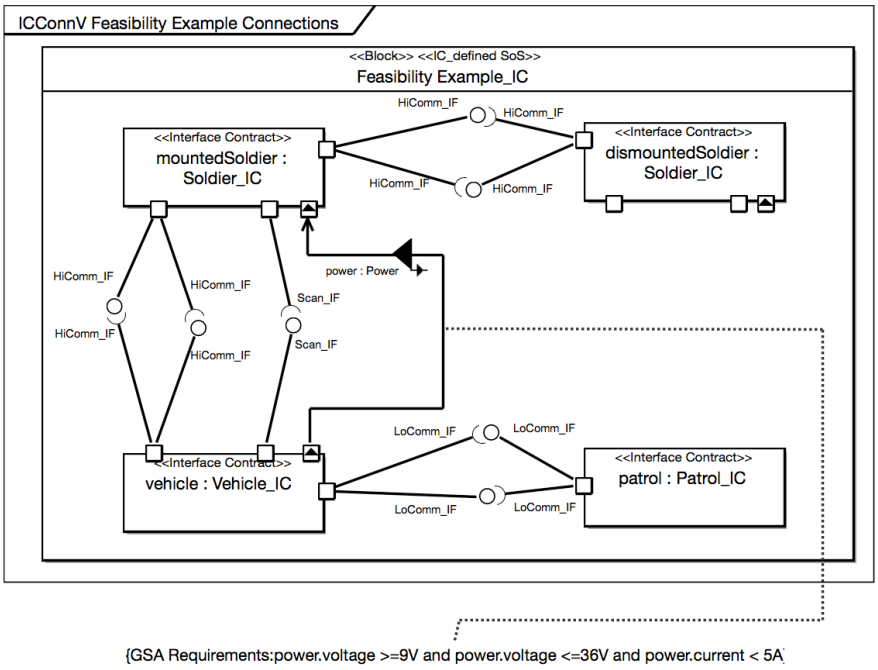

Figure 3: Interface Contractual Connections Viewpoint including power flows and constraints.

4.3 CML and Simulation

Given a semi-formal, diagrammatic, model of the example SoS architecture, we consider the use of formal notations, which afford the use of formal analysis. In this study, we propose the use of CML (Woodcock et al. 2012). This notation is based on well-established formal languages (VDM (Fitzgerald et al. 2005) and CSP (Hoare 1978)) with a formal mathematical semantics. The language allows the modelling of data, functionality, event ordering and communication, and is extensible. The language, developed in the COMPASS project, has an associated tool platform, Symphony[10], containing a broad range of formal analysis techniques. Prototype tool support has been developed for translating models from SysML into CML.

Symphony provides a collection of analysis tools to validate and verify a CML model:

- the *Interpreter* and *RT-Tester* used for requirement validation,
- the *Theorem prover* and *Model checker* may be used for property verification,
- the *Static Fault Analysis* plugin allows FMEA and fault tree analysis,
- and external links allow distributed SoS engineering.

Simulation can be partially automated by developing CML processes that embody specific tests, and running these in parallel with the system under test. This allows us to conduct both positive and negative tests (intended to test the absence of incorrect behaviours), and these processes can be derived directly from the views of the IC pattern.

4.4 Fault Modelling

One possible means of achieving dependability is for a system to tolerate faults. In SoS, this is a nascent area of research. The COMPASS project developed a Fault Modelling Architectural Framework (or FMAF) which allows engineers to consider how the different elements in an

[10] http://www.symphonytool.org/

SoS may fail and identify possible consequences of those failures. The FMAF also allows an engineer to design redundancy into a SoS and consider recovery strategies.

The FMAF comprises viewpoints for: identifying faults, errors and failures; defining causal links; and viewpoints for erroneous and recovery behavior. In this paper, we show a Threats Chain View in Figure 4. The figure shows a fault may be located in the *mountedSoldier Soldier_IC*, which may cause an error detected by that soldier which, if not recovered from, may cause the SoS failure. The remainder of the FMAF is applied in (Payne et al. 2015).

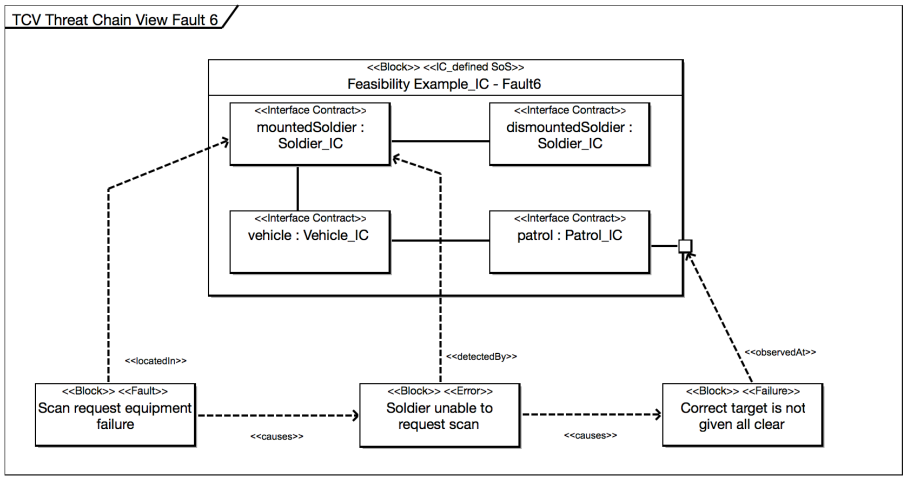

Figure 4: Threat Chain View for scan equipment failure

4.5 SoS-ACRE

Understanding stakeholder roles, requirements and the stakeholder perspectives on those requirements is a key concern in SoS engineering in LOSA. In addition, the ability to trace requirements from their source, through to model elements and analysis results is also of importance in complex SoSs. We consider the use of SoS-ACRE (Holt et al. 2012), the SoS Approach to Context-based Requirements Engineering, to help understand requirements and the contexts in which they may be considered.

The SoS-ACRE technique proposes the identification of the different CSs of the SoS, and the identification of stakeholders. Having identified the study requirements, CSs and stakeholders, we may place the requirements in context. SoS-ACRE advocates analysing the requirements in the context of each separate CS, and then combining them on a separate, SoS-level view, the Context Interaction View. This is seen as a key part of SoS requirements engineering by providing an expanded view of requirements, presented in their separate contexts, for the entire SoS. This may be the first time that requirement and contexts are analysed together, and is helpful for identifying conflicts, inconsistencies and duplicated functionality.

The SoS requirements are analysed in the context of the different CSs in Figure 5. In these diagrams, requirements are represented as use cases, and the CSs as ‘users’. Where a CS has some involvement or interest in a requirement, there is a link between the user and a use case.

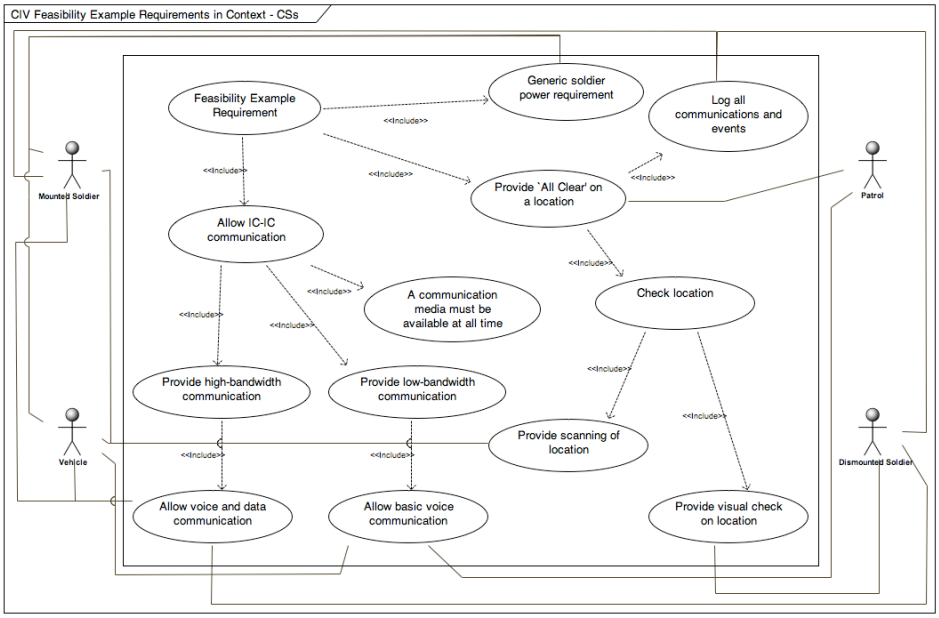

Figure 5: Context Interaction View of requirements – CSs

4.6 Co-modelling and Co-simulation

In a co-model, we model both the software and hardware elements of the LOSA SoS. The software elements are modelled using a *discrete-event* (DE) notation (we use VDM-RT), and hardware using *continuous-time* (CT) notation (such as differential equations). In simulation, DE models only represent points in time at which the state changes, and provide good abstractions for software (e.g. data types, object-orientation, threading). However, they are less suited for physical system modelling. In contrast, in CT model simulation the state changes continuously through time. CT models have good abstractions for physical system disciplines (e.g. mechanical, electrical and hydraulic system elements) but typically offer poor support for modelling software. Using co-modelling techniques, we can define both DE and CT models and define a contract to allow them to be simulated together. We use the Crescendo tool[11] for co-modelling; taking advantage of the Overture tool[12] and 20-Sim[13] for DE and CT modelling respectively. We model a LOSA SoS with one vehicle and two mounted soldiers drawing power from the vehicle. The power use from the two soldiers changes over time.

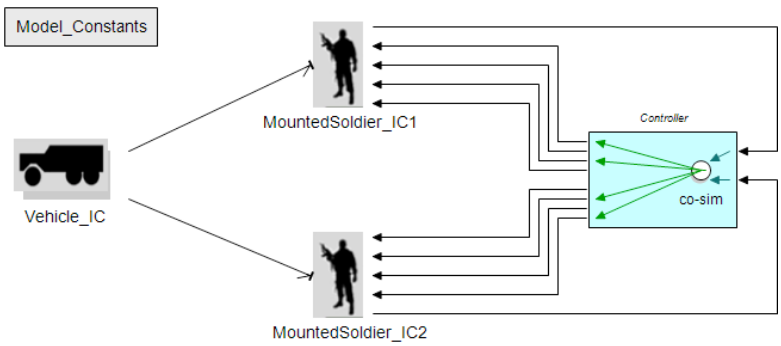

Figure 6: SoS-level 20-sim block diagram connected by signals and energy bonds

[11] http://www.crescendotool.org/
[12] http://www.overturetool.org/
[13] http://www.20sim.com/

Having defined the DE and CT models and the co-model interface, the models can be simulated. We use the Crescendo tool to enable this co-simulation. We encode a simple scenario in the DE model which uses the different mounted soldier function modes over time. The scenario can be summarised using the graphical output in Figure 7. In the chart titled MS1_ScanFunction (lhs of Figure 7) the power used changes as the Scan function changes mode: *predicting coordinates (175W)*; *turned off (0W)*; *image processing (250W)* and *reviewing playback (125W).*

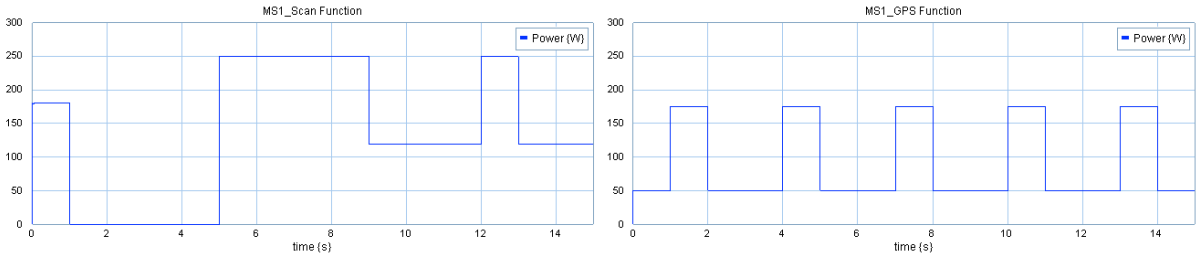

Figure 7: Co-simulation results

In Figure 7, in the chart labelled MS1_GPSFunction, the function changes from *standby mode (50W)* to a *relay position mode (175W)* periodically. This corresponds to the GPS unit sending the position to some central command on a regular basis.

Given these scenario-based power uses, the 'complex' controller of the battery charger will use power to charge the soldier's battery (300W) when it is appropriate to do so. In this particular controller policy, we state that the battery may be charged when the total power being consumed by other functions is less than 200W. This results in a power usage as shown in Figure 8, in the chart labelled MS1_BatteryCharge.

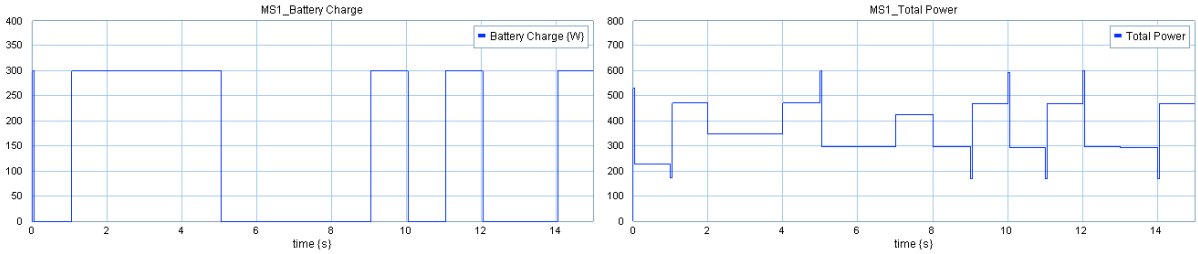

Figure 8: Co-simulation results

In the chart labelled MS_1TotalPower in Figure 8, shows the resultant total power usage. This chart displays the consequences of the different controllers in terms of the total power used by the mounted soldier. The design decisions made result in a potentially unsatisfactory situation, where power spikes are produced where the power being drawn temporarily increases to 600W, until the battery controller reacts and stops charging the battery.

This is a powerful result in that the software designer may better understand the consequences of policy design at an early stage of development and may choose a different strategy; for example using forecasting, shorter sensing periods, or some distributed control algorithms.

5. Conclusions

In this study, we identified and assessed the use of several modelling technologies. This assessment is to establish the feasibility of a pragmatic method of assessing safety and reliability dependencies of an SoS in the LOSA context. The result of this study is that such a method *is* feasible. In this section, we provide a brief assessment of their suitability for dependability assessment and in the LOSA context.

	Availability: what is the behaviour if HiComm is unavailable?	Availability: what is the behaviour if power goes below 9v?	Reliability: If a LoComm message goes missing, will mission logs remain equal?	Reliability: If power drops below 9v, will mission logs remain equal?	Safety: Can we distinguish safe and unsafe states?
Architectural Modelling	Model when functions are available and analyse with behavioural views	Changes in power can be modelled as conditions within the model	Can model lost messages and analyse with behavioural views	Changes in power can be modelled as conditions within the model	Can distinguish safe and unsafe states
Formal Modelling - Simulation	Can explore the consequences of function interfaces not being available	Not suitable (discrete values only)	Explore equality of Mission Logs during and after mission	Not suitable (discrete values only)	Check for runtime violation of invariants
Formal Modelling - Model-checking	Answer questions formulated for yes/no answers	Not suitable (discrete values only)	Answer questions formulated for yes/no answers	Not suitable (discrete values only)	Decide if certain states are reachable or unreachable
Formal Modelling - Theorem-Proving	Prove or disprove specific theorems	Not suitable (discrete values only)	Prove or disprove specific theorems	Not suitable (discrete values only)	Decide if certain properties hold
Fault Modelling	Explore detection of unavailability faults and manifestation as a failure	Less suitable – model as discrete values and examine consequences of value changes	Explore detection of faulty communication and consequences in the SoS	Less suitable – model as discrete values and examine consequences of value changes	Suitable for modelling consequences of reaching unsafe states
Co-modelling	Not suitable	Can model power as continuously changing variable	Not suitable	Can model power as continuously changing variable	Not suitable
Co-simulation	Not suitable	Explore interaction between discrete and continuous aspect of combined model	Not suitable	Explore interaction between discrete and continuous aspect of combined model	Not suitable

Table 2. Techniques/ dependability properties matrix

5.1 Dependability

In Table 2 we list the five dependability-related questions across the top, and the various techniques considered in this feasibility study on the left-hand side. Broadly speaking, the discrete techniques (the first five) are most applicable to the first three questions (columns 1, 3

and 5) and the co-modelling/co-simulation techniques are most applicable to the questions that involve power considerations.

Using the different discrete techniques (architectural modelling, formal modelling, formal analysis and fault modelling), we can model systems being unavailable, unreliable communications media and distinguish between safe and unsafe states. Using simulation of a formal model, we may explore the consequences of a LOSA SoS which is not dependable. Fault modelling allows us to explore the threats to dependability in a SoS and understand the relationships between faults in a SoS (due to CSs being unavailable/unreliable) and the failures which may occur as a consequence.

Co-modelling and co-simulation techniques allow us to model continuous variables – relating to power, for example – in notations that are able to represent continuously changing values, and explore the interaction between digital and physical phenomena. This technique allows us to consider dependability and fault tolerance in a network-connected SoS in which both data and power are present and interact.

5.2 Modelling in LOSA

Model-based Ontology The model-based ontology using an Ontology Definition View has proved to be a valuable tool. We were able to clarify and formally define the relationships between architectures, standards and implementations. This clarity proved useful throughout the project and was a useful communication tool. Although by itself an ontology does not allow the assessment of SoS dependencies, it formed a basis for the rest of the work.

Context-based Requirements Engineering We demonstrated a requirements engineering technique that allows requirements to be placed in the context of SoSs – SoS-ACRE. Although the technique is somewhat cumbersome, top-level requirements can be broken down into their implications for individual stakeholders and these can be analyzed visually for any conflicts between stakeholders. Identifying such conflicts is can help avoid the failure to meet SoS-level requirements. Model-based requirement engineering techniques also allow trace links from requirements through to model elements. This provides the possibility of developing methods for dependability-specific requirement engineering.

Architectural Modelling: Interface Contract Pattern We demonstrated the use of semi-formal modelling technologies – namely the Interface Contract pattern in SysML. This provided many advantages: consistency checking of the defined model allows for rigorous, consistent models and in general semi-formal (diagrammatic) models have good potential as a communication mechanism and are readily understandable by engineers. Furthermore, the (potentially automated) link to formal notations provide the option for advanced analysis techniques such as model checking and theorem proving. Whilst the SysML tool support is mature, the IC technique has been demonstrated only using pilot studies. Further investigation is needed to take this to a higher TRL. The architectural modelling allowed the assessment of the dependability of the LOSA SoS using model review and exploration. In our case this review was conducted informally.

Model Analysis We demonstrated a formal version of the feasibility example using CML, and simulated its behavior used the Symphony tool, in order to ensure that SoS was capable of executing the scenario, and to increase our confidence in the original design. Whilst formal techniques are improving, they are not of high enough TRL for deployment in their "raw" form. Full integration with some established modelling techniques (e.g., SysML) is vital, in order to take advantage of systematic analysis of dependability properties. Furthermore, they are

typically designed for and applied in discrete domains. We did not apply model checking or theorem proving here.

The Fault Modelling Architectural Framework (FMAF) helps the SoS engineer to understand "what if" scenarios in the same modelling framework as the architecture models produced with the IC pattern. This is due to the fact that the FMAF has been incorporated into the SoS engineering approach, via a fault modelling SysML profile. The FMAF proved valuable in identifying and managing causal chains leading to potential system and SoS failures, in understanding the consequences of dependability threats and in designing recovery strategies.

Co-modelling allows engineers from different domains to model an SoS and the CSs in their own notations and, along with co-simulation, enables trade-off analysis, design space exploration and fault modelling. This project used VDM-RT for modelling software, 20-Sim for modelling physical systems and the Crescendo tool for co-simulation. The technology used in this study is used primarily in embedded and cyber-physical system modelling. Fully incorporating it into an SoS engineering approach will require more effort, with multi-tool chain and diverse notations suitable for different stakeholders.

The co-modelling and co-simulation techniques have the potential to aid analysis and assessment of dependability properties, especially in contexts which integrate discrete and continuous domains. This is of great importance within LOSA, where the CSs of the SoS are connected with both data and physical interfaces.

6. Future Work

The SoS and CPS engineering techniques we have shown form a feasible basis for a pragmatic approach to assessing reliability dependencies, although the different elements are at different levels of maturity. Further investigation would be needed to incorporate the different technologies into different frameworks for use in LOSA. For example, it may be possible to incorporate the approach into MODAF. This would be helped by the already consistent definitions and use of viewpoints and ontology. We outline areas of future work in this section.

1. We have concentrated on technologies produced in the COMPASS and DESTECS EU projects. Further study will compare these model-based technologies with potential alternatives. This comparison must also take into account cost-effectiveness and usability dimensions, and will aid in understanding the impact of incorporating new technologies in normal SoS engineering practice in LOSA.
2. An assessment is required to determine the potential to integrate with relevant engineering processes and standards (such as MODAF) in the defence industry and to input to future LOSA and COI(L) standards and LOSA joining rules.
3. The modelling techniques investigated used a simple feasibility example and in cases, notably requirements engineering, was only able to provide a partial treatment of the technologies. A full treatment of stakeholder requirements modelling including requirements that span both DE and CT elements of a LOSA SoS is required.
4. The co-modelling technologies assessed do not provide a contractual style of modelling as demonstrated in the architectural modelling techniques, deemed useful in SoS engineering. An integrated method for contractual modelling and co-modelling in LOSA would be valuable. It would allow a unified approach to representation and simulation of models of systems which have both software and physical properties at the SoS-level.
5. Several methodological approaches are available with co-modelling, including design-space exploration (DSE). The application of DSE in LOSA should be considered to

aid in understanding power requirements over the life time of system, and may be useful for maintenance and cost models of LOSA SoS.

6. The various modelling technologies provide analysis results in the form of simulations, semi-formal model review, formal analysis and co-simulation. These results need to be placed in the context of safety cases in LOSA to determine how model-based techniques may be used as evidence in safety and dependability cases.

Acknowledgments

This paper presents the findings of a project funded by DE&S Technology Office. The authors also acknowledge the support of the UK EPSRC platform grant on Trustworthy Ambient Systems (TrAmS- 2), and by the INTO-CPS project funded by the European Commission's Horizon 2020 programme under grant agreement number 664047.

References

Andrews, Z. Ingram, C. Payne, R. Romanovsky, A. Perry, S. and Holt, S. 2014 "Traceable engineering of fault tolerant SoSs." In INCOSE IS 2014.

Avizienis, A. Laprie, J-C. Randell, B. and Landwehr, C. 2004 "Basic Concepts and Taxonomy of Dependable and Secure Computing." *IEEE Transactions on Dependable and Secure Computing*, 1:11–33.

Benveniste, A., Caillaud, B., Nickovic, D., Passerone, R., Raclet, J., Reinkemeier, P., Larsen, K. (2012). "Contracts for Systems Design". INRIA report 8147.

Broy, M. 2013 "Engineering Cyber-Physical Systems: Challenges and Foundations." In *Complex Systems Design & Management*, pages 1–13. Springer Berlin Heidelberg.

Bryans, J. Fitzgerald, J. Payne, R. and Kristensen, K. 2014 "Maintaining Emergence in Systems of Systems Integration: a Contractual Approach using SysML". In INCOSE International Symposium (IS2014).

Bryans, J. Fitzgerald, J. Payne, R. Miyazawa, A. and Kristensen, K. 2014 "SysML Contracts for Systems of Systems." In System of Systems Engineering Conference, Adelaide.

Cantot, P. AND Luzeaux, D. 2011. Simulation and Modeling of Systems of Systems. Wiley

Coleman, J. W. Couto, L. D. Lausdahl, K. Nielsen, C. B. Malmos, A. K. Larsen, P. G. Payne, R. Foster, S. Miyazawa, A. Schulze, U. Cajueiro, A. and Didier, A. 2014 "COMPASS tool user manual." Technical report, COMPASS Deliverable, D31.4a.

Damm, W., Hungar, H., Josko, B., Thomas, P., & Stierand, I. (2011). "Using contract-based component specifications for virtual integration and architecture design". In *Design, Automation Test in Europe Conference Exhibition*

Dahmann, J. 2014 "Systems of Systems Pain Points." In INCOSE IS 2014

Defence Committee UK. 2013 "Defence Acquisition". 7th Report of Session 2012-13, Vol 1.

DoD, 2008. "Systems and Software Engineering. Systems Engineering Guide for Systems of Systems. Tech. Rep. Version 1.0.", Office of the Deputy Under Secretary of Defense for Acquisition & Technology, Department of Defense.

Fitzgerald, J. and Larsen, P.G. 2009 "Modelling Systems – Practical Tools and Techniques in Software Development." Cambridge University Press, The Edinburgh Building, Cambridge CB2 2RU, UK, Second edition. ISBN 0-521-62348-0.

Fitzgerald, J. Larsen, P.G. and Verhoef, M. editors. 2014 “Collaborative Design for Embedded Systems – Co-modelling and Co-simulation.” Springer.

Holt J. and Perry, S. 2014 “SysML for Systems Engineering: A model-based approach.” IET, 2nd edition: 2014.

Holt, J. Perry, S. Brownsword, M. Cancila, D. Hallerstede, S. and Hansen, F. 2012 “Model-based requirements engineering for system of systems.” In System of Systems Engineering (SoSE), 2012 7th International Conference on, pages 561–566.

Holt, J. Perry, S. Hansen, F. O. Hallerstede, S. Kristensen, K. and Riddle, S. 2014 “Final Report on Guidelines for Architectural Level SoS Modelling.” Technical report, COMPASS Deliverable, D21.5. Available at http://www.compass-research.eu/.

INCOSE, 2010 “Systems Engineering Handbook. A Guide for System Life Cycle Processes and Activities.” INCOSE, 7670 Opportunity Rd., Suite 220 San Diego, CA.

INCOSE, 2014. “A World in Motion; Systems Engineering Vision 2025”

Ingram, C. Payne, R. Fitzgerald, J. Cuoto, L.D. 2015 “Model-based Engineering of Emergence in a Collaborative SoS: Exploiting SysML & Formalism” INCOSE IS 2015.

Kopetz, H., Hoftberger, O., Fromel, B., Brancati, F., & Bondavalli, A. 2015 “Towards an understanding of emergence in systems-of-systems.” In System of Systems Engineering Conference (SoSE), 2015 (pp. 214–219)

Meyer, B. (1998). Object-Oriented Software Construction (2nd ed.). Prentice-Hall.

Maier, M.W. 1998 “Architecting principles for systems-of-systems.” Systems Engineering, 1(4):267–284.

MOD UK. 2012a “MOD Architecture Framework”. Technical report, Ministry of Defence, UK. Available at https://www.gov.uk/mod-architecture-framework.

MOD UK 2012b “National Security Through Technology”. Technical report Cm8278, Ministry of Defence, UK

MOD UK. 2013a “Generic Soldier Architecture.” DefStan 23-12, Ministry of Defence, UK.

MOD UK. 2013b “Generic Vehicle Architecture.” DefStan 23-09, Ministry of Defence, UK.

Nielsen, C.B., Larsen, P.G., Fitzgerald, J., Woodcock, J., Peleska, J. 2015, “Systems of Systems Engineering: Basic Concepts, Model-Based Techniques, and Research Directions.” *ACM Computing Surveys*. 48(2): Article 18 (September 2015), 41 pages.

OMG. 2012 “OMG Systems Modelling Language Version 1.3.” Available: http://www.omg.org/spec/SysML/1.3 (Accessed June 2013)

Payne, R. Bryans, J. Fitzgerald, J. 2015 “System of Systems Dependability for LOSA” Newcastle University School of Computing Science Technical Report CS-TR-1487 Available: http://www.compass-research.eu/Project/Publications/CS-TR-1487.pdf

Perry, S. Holt, J. Payne, R. Miyazawa, A. Woodcock, J. Hansen, F.O. Nielsen, C.B. Iyoda, J. and Cornelio, M. 2012 “Initial report on SoS architectural models.” Technical report, COMPASS Deliverable, D22.1. Available at http://www.compass-research.eu/.

Woodcock J. and Miyazawa, A. 2012 “Features of CML: a Formal Modelling Language for Systems of Systems.” In Proceedings of the Conference on System of Systems Engineering (SOSE 2012), Genoa, Italy. IEEE.

Biography

Dr John Fitzgerald is a full Professor in Computing at Newcastle University, UK, where he heads the Cyber-Physical Systems Lab. He studied verification technology (PhD, Manchester Univ., 1991) before researching avionics design with BAESYSTEMS as a SERC Fellow and Lecturer at Newcastle. He has contributed to model-based methods and tools applied commercially in areas as diverse as firmware design and options trading. He led the EU's project on comprehensive modelling for advanced systems of systems (COMPASS), and now leads research on methods for co-modelling and co-simulation in the INTO-CPS project. He is academic lead for Computing in Newcastle University's $90m centre for digitally-enabled urban sustainability. He is a member of INCOSE and a Fellow of the BCS.

Dr. Richard Payne is a Research Associate in the School of Computing Science at Newcastle University, UK. Following his PhD (Newcastle University 2012) on models of dynamic reconfiguration, he worked on projects in architectural modelling in Systems of Systems (SoSs). He is currently working on SoS and Cyber-Physical System (CPS) projects, including the EU H2020 INTO-CPS project which is aiming to produce a model-based tool chain for CPS design and development. His research interests include architectural modelling, dynamic architectural reconfiguration, the pragmatic application of formal modelling, and SoS/CPS engineering.

Dr. Jeremy Bryans is a Research Fellow in the Centre for Mobility and Transport at Coventry University, UK. His research concerns the modelling and analysis of collaborating systems, and in the development of trustworthy policies for their safe and secure interaction. Following a PhD in formal semantics from Reading University he has worked in Canterbury, Stirling and Newcastle University. Recently he worked on the COMPASS project, where he worked on the semantic foundations for a modeling language for Systems of Systems, as well leading projects on Smart Grid Infrastructure modelling and Trustworthy Dynamic Coalitions. He is a member of INCOSE and the BCS.

Elspeth Winthorpe is a member the MOD's Defence Science Engineering Graduate (DESG) scheme, with a keen interest in System of Systems and Cyber Physical Systems and how they can be applied to Defence. As part of the DESG scheme she recently completed masters in Security and Resilience at Newcastle University UK, as part of this Elspeth completed a dissertation, expanding the work of Newcastle University on LOSA component Generic Base Architecture, approaching it from both a Cyber Physical Systems and Systems of System perspective.